2020-05

Understanding and Modeling the Life-Cycle Cost Tradeoffs Associated with the Procurement of Open Systems

Sandborn, Peter; Chen, Shao-Peng; Lucyshyn, Willam

Monterey, California. Naval Postgraduate School

SYM-AM-20-069

PROCEEDINGS
OF THE
SEVENTEENTH ANNUAL
ACQUISITION RESEARCH SYMPOSIUM

Acquisition Research:
Creating Synergy for Informed Change

May 13–14, 2020

Published: April 15, 2020

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

The research presented in this report was supported by the Acquisition Research Program of the Graduate School of Defense Management at the Naval Postgraduate School.

To request defense acquisition research, to become a research sponsor, or to print additional copies of reports, please contact any of the staff listed on the Acquisition Research Program website (www.acquisitionresearch.net).

Understanding and Modeling the Life-Cycle Cost Tradeoffs Associated with the Procurement of Open Systems

Shao-Peng Chen—is a Research Assistant in the Department of Mechanical Engineering at the University of Maryland, College Park. Chen's research interests include life-cycle modeling and system reliability. He received his BS in mechanical engineering from National Taiwan University. He is currently pursuing a PhD in mechanical engineering at the University of Maryland. [spchen@umd.edu]

Peter Sandborn— is a Professor in the Department of Mechanical Engineering and the Director of the Maryland Technology Enterprise Institute (Mtech) at the University of Maryland. Sandborn's research interests include system sustainment, electronic part obsolescence management, prognostics and health management for electronic systems, and system life-cycle and risk economics. He received a PhD in electrical engineering from the University of Michigan. [sandborn@umd.edu]

William Lucyshyn—is a Research Professor and the Director of Research at the Center for Public Policy and Private Enterprise, in the School of Public Policy, at the University of Maryland. Lucyshyn's research interests include policy issues related to public sector management and operations, how government works with private enterprise, public and private sector partnering, transforming DoD logistics and supply chain management, and government sourcing and acquisition best practices. Lucyshyn received his BS in engineering science from the City University of New York in 1971 and MS in nuclear engineering from the Air Force Institute of Technology in 1985. He was certified Level III as an Acquisition Professional in Program Management in 1994. [lucyshyn@umd.edu]

Abstract

Openness (of a system or architecture), though intuitively understood, remains difficult to quantify in terms of its value. Although commonly associated with cost avoidance, system openness can also increase costs. Previous efforts have relied on highly qualitative system analyses, with the results often articulated as an intangible "openness score," for determining which of multiple system implementations is more open. Such approaches do not provide enough information to make a business case or understand the conditions under which life-cycle cost avoidance can be maximized (or whether there even is cost avoidance). This paper develops a multivariate model that quantifies the relationship between system openness and life-cycle cost. A case study that evaluates the utility of the second phase of the Acoustic Rapid COTS Insertion (A-RCI) Sonar System's evolution to an open system is discussed.

Keywords: Open systems, Cost modeling, Obsolescence management, A-RCI

Introduction

The United States faces several long-term budgetary challenges. The rising costs of mandatory entitlement programs, coupled with the budget deficits projected into the foreseeable future, create inevitable downward pressure on future Department of Defense (DoD) budgets. As an example, the DoD's 2020 budget request for acquisition was $247 billion (about one-third of its total budget request): $143 billion for procurement and $104 billion for research, development, test, and evaluation (RDT&E). This amount is approximately 1% less than the amount appropriated for 2019 when adjusted for inflation. Under the 2020 Future Years Defense Program (FYDP), this downward trend is expected to continue through 2024, decreasing the budget to $230 billion adjusted for inflation. This 7% decrease will constrain the funds available for recapitalization, modernization, and transformation of the military (Congressional Budget Office [CBO], 2019). Future DoD budgets will require hard decisions and force a reengineering of processes and the most efficient use of resources.

DoD weapon systems have historically been developed using acquired proprietary systems and interfaces. These make it challenging to modernize and reduce opportunities for

competition. For example, the Air Force's program to upgrade the B-2 bomber's communications, networking, and defensive management will cost over $2 billion, since the prime contractor owns all the necessary proprietary technical data and software. Completing this effort was not a viable financial option (Government Accountability Office [GAO], 2014).

A variety of strategies are being explored or reemphasized to increase the efficiencies of acquisition processes. One way for the DoD to minimize the cost and time needed to modify or upgrade weapon systems is by using a modular open systems approach (MOSA) for system design and development. When used appropriately, MOSA provides a degree of flexibility, enabling the integration of rapidly changing technologies. However, as with all approaches, there are costs as well as benefits. This paper explores a business case methodology to assess the cost effectiveness of MOSA.

Modular Open Systems Approach (MOSA)

System openness refers to the extent to which system components (e.g., hardware and software) can be independently integrated, removed, or replaced without adverse impact on the existing system. Current DoD policy calls for the use of modular open systems design to the maximum extent possible. In fact, the acquisition strategy for a given system must identify where, why, and how modular open systems will be used (DoD, 2015). Conventional wisdom supports the notion of open systems, but quantifying the actual cost avoidance remains elusive. The objective of this paper is to quantify the relationship between system openness and life-cycle cost.

Open Systems Architecture, now referred to as Modular Open Systems Approach (MOSA), has been widely endorsed by the DoD since the mid-1990s. MOSA promotes the use of modular design to encourage companies to improve and manufacture technology that is interoperable with the DoD's current system. Formally, the DoD defines MOSA as "a technical and business strategy for designing an affordable and adaptable system." An example of this could be the use of radar technology on an aircraft. Under MOSA, the radar technology could be replaced or upgraded without replacing the whole aircraft or numerous related subsystems. Closed systems architecture, on the other hand, effectively restricts access to configuration and programming information from outside parties. Closed systems often make upgrading a piece of equipment difficult and costly. Further, these closed systems can lead to vendor lock. where the DoD becomes dependent on a single service provider because the costs of changing vendors is prohibitive.

In 1994, the Open Systems Joint Task Force (OSJTF) was established to promote the use of open systems architecture from the top-down throughout the DoD. Ever since, there has been an ongoing effort throughout the Department to widely implement open systems architecture. In May 2003, DoD Directive 5000.01 emphasized the use of "modular, open-systems approach … where feasible" (DoD, 2018). Further, in 2004, the OSJTF was identified as the DoD lead for MOSA. Soon after, OSJTF stated that MOSA would be an integral part needed to improve the DoD's joint combat capabilities.

In a January 2015 DoD Directive, program managers were further directed to use an open-systems approach wherever "feasible and cost-effective" (DoD, 2015). Specifically, program managers are to use the Acquisition Strategy to identify where, why, and how MOSA will or will not be used. Once the Acquisition Strategy document is completed, the formal project or procurement can begin. Under the "Business Strategy" section of the Acquisition Strategy, managers are instructed to complete a business case analysis calculation with the help of engineering tradeoff analysis "that outlines the approach for using open systems architecture and acquiring technical data rights" (DoD, 2015). Finally, managers must compare and analyze the

results of open architecture to closed architecture to determine which is most cost effective given the military's specific need.

In 2017, the National Defense Authorization Act (NDAA) for Fiscal Year (FY) 2017 amended and formalized the implementation of MOSA (Pub. L. 114–328; 10 U.S.C. § 2446a, 2016). Following January 1, 2019, the 2017 NDAA stated that MOSA "shall be designed and implemented, to the maximum extent possible … to enable incremental development and enhance competition, innovation, and interoperability." In January 2019, a memo from the secretary of the Navy, the secretary of the Air Force, and the secretary of the Army emphasized their commitment to MOSA, stating "further development of Modular Open Systems Approach standards in areas where we lack them is vital to our success." The most recent NDAA for FY 2020 put responsibility on the secretaries of the military departments to implement MOSA (Pub. L. 116–92; 10 U.S.C. § 840, 2019). Each branch has responded in slightly different ways to implement MOSA. Broadly, DoD leadership and Congress strongly support MOSA and hope to implement it more completely in the near future.

An open systems approach (OSA), when used in conjunction with a modular architecture, reuse, and/or the harnessing of existing technologies (commercial off-the-shelf [COTS] or proprietary), is commonly associated with cost avoidances arising from more efficient design, increased competition among suppliers, more efficient innovation and technology insertion (faster, cheaper design evolution), and the modularization of qualification. However, the high levels of investment and the increased risk exposure over long system life cycles are often overlooked. Determining if, and to what extent, openness should be pursued remains challenging in the absence of a quantitative model that elucidates the relationship between the degree of system openness and the system's life-cycle cost.

Historically, critical functionality in complex electronic systems was provided by custom-made components and custom proprietary architectures, requiring long development times and high development costs. However, recent technological advancements have allowed for the increased generalizability of both hardware and software (and system architecture); now components can be designed once, and then used in many different applications (Guertin & Miller, 1998). These advancements have increased the viability of using OSA in general and a Modular Open Systems Approach (MOSA) in particular (Abbott, Levine, & Vasilakos, 2008).

While the DoD supports implementing MOSA whenever possible, there are numerous reasons to be cautious since business and engineering-tradeoffs must be made, possibly changing the incentive structure and reducing the system effectiveness. First, if there are no standards for a new product, then closed system architecture may be best until standards are created (Firesmith, 2015). Second, a poorly designed modularized architecture may be too costly to re-architect, and it is better to stick with the old. Third, there is only one qualified vendor to provide the service, making opening the system costly without benefit. These drawbacks challenge the current DoD posture of implementing MOSA throughout the department, but neither side has successfully provided quantitative analysis to support their position.

Generally, it is taken for granted that the use of OSA and MOSA decreases the total life-cycle cost of a system. Leveraging existing open technology, including COTS components, avoids many costs associated with designing custom systems and reduces the time required for development or refresh of a system (Logan, 2004). The use of OSA or MOSA helps mitigate the effects of obsolescence, lengthens the system's support life, allows for the incremental insertion of new technologies (Open Systems Joint Task Force [OSJTF], 2004; Boudreau, 2007), and evolving functionality. The use of well-defined standards promotes smooth interfacing both within and between systems, while the proliferation of common component types fosters competition

between suppliers. Component design reuse (within and between systems) eliminates redundant components, thus reducing logistical costs.

However, there are costs associated with openness that should be considered. Building a subsystem from open standards and commercially available components often relies on the use of generalized technology with unnecessary, and costly, additional functionality, increasing the cost, complexity, and effective failure rates (Bass et al., 2008; Hanratty, Lightsey, & Larson, 2002). In other cases, it may be necessary to modify COTS components to meet performance requirements (Jensen & Petersen, 1982; Wright, Humphrey, & McCluskey, 1997), thereby adding costs. In addition, the enterprise that manages the system likely has no control over the supply chains for COTS components, which tend to be more volatile than proprietary ones (Lewis, Hyle, Parrington, Clark, Boehm, Abts, & Manners 2000). This makes it desirable to refresh open systems designs more frequently (Abts, 2002; Clark & Clark, 2007), which leads to an increase in the number of fielded configurations, which complicates logistics, resulting in more expense.

This paper seeks to quantitatively analyze MOSA, specifically the relationship between MOSA and life-cycle cost. Throughout numerous DoD documents, some form of life-cycle cost savings is cited a majority of the time as a reason to support MOSA implementation. One study cites preliminary results compiled from 10 years of data on both the Acoustic Rapid COTS Insertion program and its predecessor. These results indicate that life-cycle cost improved by nearly 5:1, but the underlying data is unavailable (Boudreau, 2006). Without quantitative analysis, DoD program managers are left without much guidance on when MOSA is best given life-cycle costs.

Existing Work

Several previous efforts have addressed the measurement of system openness. These include the MOSA Program Assessment and Rating Tool (PART), developed by the Navy's OSJTF. PART consists of a series of questions, divided into business indicators and technical indicators, that assess the extent to which a certain principle or practice is implemented. PART can be used to measure a system's openness, but it is subjective and qualitative, and the results depend on the optimism with which the survey is completed (OSJTF, 2004).

The Naval Open Architecture Enterprise Team (NOAET) developed the Open Architecture Assessment Model (OAAM) to "define, measure, and illustrate the relative levels of openness" (Naval Open Architecture Enterprise Team [NOAET], 2009). Like PART, OAAM measures the openness of a system using both business and technical characteristics, which are combined to produce an overall openness characterization. While OAAM was relatively simple, it lacked resolution, so NOAET created the Open Architecture Assessment Tool (OAAT) to expand and build upon OAAM (NOAET, 2009). OAAT uses a questionnaire that includes PART to assess system openness (NOAET, 2009).

In 2011 and 2012, several parties, including the Air Force Research Laboratory's RYM subgroup, collaborated to develop a set of metrics to evaluate the openness of an architecture. This effort focused on selecting metrics that were broad enough to assess a general case and quantifiable, so that the measurement would be repeatable. The result of this effort, called the MOSA Metrics Calculator (MOSA, 2012), improves upon the PART questionnaire by ensuring that all metrics are quantifiable.

One common attribute of the PART, OAAT, and the MOSA Metrics Calculator is that they don't account for the cost associated with an open systems approach and are unable to measure the value of the benefits obtained. They cannot be used to make a business case for the use of openness (i.e., they explicitly assume that increased openness is always beneficial, but is it?).

Another approach to measuring openness comes from PMH Systems and the University of Southampton. This work uses an easily quantifiable metric, the fraction of interfaces that use open standards, and a stochastic model to estimate the decrease in cost and development time associated with increasing openness (Henderson, 2009). However, the model implicitly relies on the assumption that increased openness is always beneficial. Additionally, the metric developed cannot resolve different levels of openness and, most importantly, only addresses the design phase, ignoring significant costs and avoidances that occur later in the system's life cycle.

Research Objectives

Previous efforts have relied on a highly qualitative analysis of a system, with the results often given as an intangible "openness score" used to determine which of multiple system implementations is more open. Such approaches do not provide enough information to make a business case or understand the conditions under which life-cycle cost avoidance can be maximized (or whether there even is cost avoidance). The objective of this paper is to quantify the relationship between system openness and life-cycle cost; the key questions the model presented can answer are

- What are the costs avoided and added due to openness?
- What are the variables that should be considered in assessing the cost impact?
- What level of openness provides the best value to the government/customer?
- How long will a given open system need to be supported to recover its initial costs?
- How should current policy be modified to ensure an appropriate level of openness is applied?

The discussion of open system pros and cons in the Introduction is by no means exhaustive, but meant to point out the complexity that is present in this problem. One possible breakdown of the total cost incurred designing, building, operating, and retiring a system is[1]

$$C_{Total} = C_{Design} + C_{Production} + C_{O\&S} + C_{Refresh/Redesign} + C_{Enterprise} \quad (1)$$

where the sections Design and Qualification Costs (C_{Design} *and* $C_{Refresh/Redesign}$) through Enterprise Costs ($C_{Enterprise}$) discuss the modeling of the various contributions to Equation 1.

The Model Development section proposes a stochastic discrete-event simulation model developed to determine the difference in life-cycle and implementation cost between two versions of the same system (having different levels of openness). The Case Study section provides a case study using the model. In the Discussion and Conclusions section, we discuss the generalization of the model to consider more general questions about the optimal openness of systems.

Model Development

This section presents a stochastic discrete-event simulation model developed to determine the difference in life-cycle and implementation cost between two versions of the same system (having different levels of openness).

Discrete-Event Simulation Model Development

The model developed follows the life history of a group of items (e.g., a bill of materials [BOM]). Life-cycle cost modeling generally involves modeling systems (more specifically, system costs) that evolve over time. For complex systems, the time-dependent costs usually involve the

[1] Some preliminary work done formulating a model based on the Equation 1 appears in Schramm (2013).

operation and support of the system. Depending on the type of system, operation may involve the purchase of fuel, the training of people, the cost of various consumable materials, and maintenance. Maintenance costs that occur over time are combinations of labor, equipment, testing, and spare parts. If the life cycle of a system is relatively short (i.e., less than a couple of years), then direct calculation methods of the life-cycle cost are applicable. However, when the modeled life cycle extends over significant periods of time and the cost of money is non-zero, the calculation of life-cycle cost changes from a multiplication problem into a summation problem, and the dates of cost events become important (e.g., the cost of individual maintenance events differ based on when they occur due to the cost of money). Discrete-event simulation is commonly used to model life-cycle costs that are accumulated over time when time spans are long and the cost of money is non-zero.

Discrete-event simulation (DES) is the process of codifying the behavior of a complex system as an ordered sequence of well-defined events. In the context of life-cycle cost modeling, an event represents a particular change in the system's state at a specific point in time, and the change in state generally has cost consequences. Discrete-event simulation utilizes a mathematical/logical model of a physical system that portrays state changes at precise points in simulated time, called events. Discrete means that successive changes are separated by finite amounts of time, and by definition, nothing relevant to the model changes between events. In DES, the system "clock" jumps from one event to the next, and periods between events are ignored. A timeline is defined as a sequence of events and the times that they occur.

At each event, various properties of the system can be calculated and accumulated. Probability distributions are used to represent the uncertainty in the simulation parameters in discrete-event simulation. This means that we simulate the timeline (and accumulate relevant parameters) with many trials (i.e., through many possible time histories) in order to build a statistical model of what will happen in the life history of the system.

In the model developed for this paper (Figure 1) the events of interest are maintenance events, production events (delivery of new systems), retirement events (retirement of fielded systems), logistics events (management of spares, lifetime buys of parts to manage obsolescence) and design redesigns/refreshes. The accumulated properties of interest are cost. The model effectively generates lists of events (date and type of event). The event dates are determined by sampling time-to-failure (TTF) distributions, forecasted obsolescence data distributions, and a predetermined refresh/redesign schedule. The event lists are then pushed through a cost model to accumulate the discounted life-cycle cost.

The description in this section is of the base model. Openness is differentiated by variations in the system components and architecture (and their associated TTF and obsolescence date distributions). The base model does not constitute the entire model. We must also model the relative design and qualification overhead (costs) associated with varying the bill of materials and system architecture (see the next subsection).

Design and Qualification Costs (C_{Design} *and* $C_{Refresh/Redesign}$)

C_{Design} are the costs associated with designing a new system that satisfies a set of requirements and include the majority of the costs incurred before the final design is selected, including the cost of designing the system, as well as the costs of partial or alternative designs considered but not implemented.[2] Prototyping and any design overhead costs are also included

[2] (i.e., not used for the current system). The ability to repurpose a previously designed subsystem for a new application is an enterprise-level cost avoidance strategy (included in $C_{Enterprise}$).

in the design costs. C_{Design} also includes the Non-Recurring Engineering (NRE) costs, which may include intellectual property costs and testing and qualification costs to demonstrate that the standards, components, subsystems, and complete system meet the required design parameters for performance, reliability, security, etc. These costs may be significantly lower for enterprises that maintain a library of previously used and qualified hardware and software components.

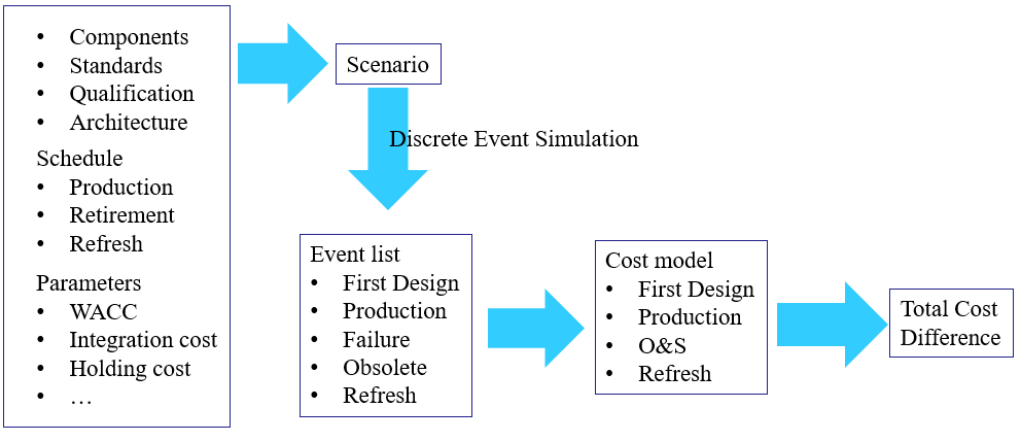


Figure 1. Cost Model Used in This Analysis

The initial system design represents the effort associated with design and qualification of the whole system. This event only occurs at the beginning of a life cycle or when a system redesign that transitions the system architecture occurs. The cost of the first design event includes design cost and qualification cost.

Design cost is the cost to design or pick the components and modules for the system. A COTS component has zero design cost, while a proprietary component has a design cost determined by the user. For a module, the design cost is related to the effort required to integrate all subcomponents in the module and is proportion to the number of subcomponents. The total design cost is the sum of every component/module design cost.

Qualification cost is the cost to conduct qualification tests for the whole system. Every component and module are assigned required qualification tests. The sum of the qualification cost of all components/modules is the qualification cost.

After the system has been in use for some period of time, it may be desirable (or necessary) to update or refresh the system to ensure it is still manufacturable and supportable. Redesigns that evolve the system functionality and performance may also occur. Refresh costs, $C_{Refresh/Redesign}$, may be similar to those initially incurred in C_{Design}, except that there is a greater opportunity for design reuse, not only of some components or subsystems.

Production Costs ($C_{Production}$)

$C_{Production}$ includes all costs to manufacture and deliver the system, including component procurement, screening and/or burn-in of hardware components, assembly/manufacturing, and any recurring testing costs.

Production events are defined as the action of purchasing components and the resources required to assemble (and functionally test) the system. The production event dates are determined by the production schedule.

Operation and Support Costs ($C_{O\&S}$)

All operation and support costs, including those associated with maintenance, sparing, obsolescence mitigation, and lack of system availability, fall under $C_{O\&S}$.

Maintenance (O&S) events occur when there is a component failure. System maintenance includes the labor to conduct the process and the cost to obtain a new component. The model assumes that the system is fixed instantly so there is no downtime associated with a failure. The model also assumes that the maintenance strategy is good-as-new component replacement. If the component is still available in the market, it would be procured as needed.

When the obsolescence of a component occurs, sufficient components are procured and held in inventory in order to support the system until the end of support of the system or the next system refresh (i.e., a lifetime or bridge buy is made)—these components must cover future production and maintenance needs. The cost incurred at the obsolescence event is the procurement cost of the components. Inventory (holding) costs are charged when the parts are taken from the inventory and used for maintenance.

Refresh/Redesign Costs ($C_{Refresh/Redesign}$)

After the system has been in use for some period of time, it may be desirable (or necessary) to update or refresh the system to ensure it is still manufacturable and supportable. Redesigns that evolve the system functionality and performance may also occur. Refresh costs, $C_{Refresh/Redesign}$, may be similar to those initially incurred in C_{Design}, except that there is a greater opportunity for design reuse, not only of some components or subsystems, but of the overall system architecture as well.

At a system refresh/redesign event, every component/module in the architecture is examined, and some components/modules are refreshed due to obsolescence or an update requirement. In the present model, system refresh events are determined before the system life cycle starts.[3] The cost of refresh, including the cost of redesign and requalification, is based on the number of components and modules needed to be refreshed, which is the combination of the obsolescence components and the other affected modules/components due to ripple effects. The refresh cost of a component/module is equal to the design and qualification cost described in the first design event.

If a component is obsolete when a refresh or redesign is encountered, it is refreshed by replacing it with another component that has the same function (assumed to be not obsolete). In an architecture where components and modules are linked to each other, one obsolete component might affect other connected components and modules, causing them to need to be refreshed as well. When the original obsolete component is replaced by a different component with the same function, it is possible that the new one would have a different way to integrate with the surrounding components/modules. Thus, those surrounding components and modules would require redesign and requalification. Further, those refreshed components and modules would affect other connected components and modules, causing a ripple effect. Alternatively, if the component connects to its surrounding components/modules via an open standard, then the component may only require a drop-in replacement that conforms to the open standard. In other words, as long as the standard is not obsolete, the ripple effect would stop when it encounters an open standard.

[3] Models exist that can determine the optimal distribution of refreshes (e.g., Singh and Sandborn [2006]).

Enterprise Costs ($C_{Enterprise}$)

$C_{Enterprise}$ are the costs incurred at the enterprise level and are shared across all of the different types of systems created by the enterprise, which includes the costs to maintain a component library or any support infrastructure that is shared by more than one system or program.

Comparing Open and Closed Systems—Relative Cost Models

It is often not practical to calculate the absolute value of all the life-cycle costs for a system—this would may be a nearly impossible task. To assess the cost of openness, we are only actually interested in the difference in the costs previously mentioned between two system implementations or architectures. This approach is referred to as a relative cost model (Sandborn, 2017). The advantage of a relative cost model is that all the costs that are a "wash" between the two architectures (i.e., the same) and don't have to be modeled because they simply subtract out. The cost difference between two cases is significantly easier to determine than the absolute cost of each of the cases. The proposed model never produces absolute costs, only cost differences between two cases.

Figure 2 shows a sample result generated using the model (the data used in this figure will be described in ARC-I Case Input Data). On the left side of Figure 2 are the cumulative discounted life-cycle costs for A-RCI1 and A-RCI2. In both cases, many time-history solutions are shown (each is the result of a unique combination of samples from the input probability distributions, so each is one possible time history for the system). The darker solid blue line is the mean of the time histories. The life-cycle costs for the two system implementations are NOT particularly meaningful because lots of relevant life-cycle costs are left out of the model. The right side of Figure 2 shows the difference between the two system costs during the life cycle. The difference between the two cases is meaningful if the costs left out of the model are a "wash" (i.e., if they approximately subtract out).

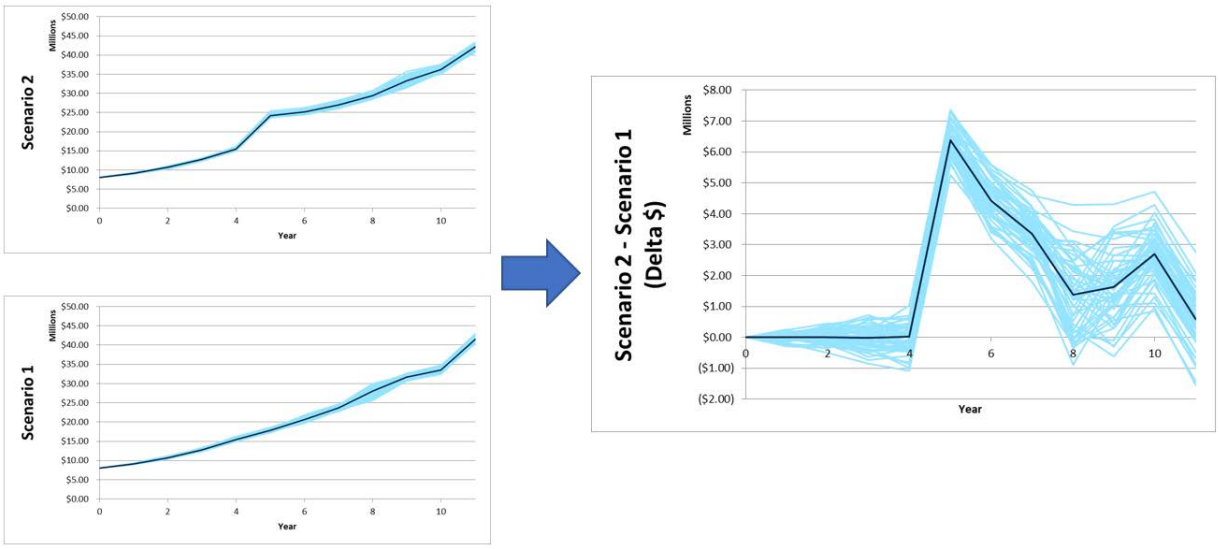


Figure 2. Sample Solution Construction of Cumulative Cost Difference. The cumulative cost difference at year 11 between the two scenarios is approximately $0.8 million.

Case Study

In this section, we present a case study of the Acoustic Rapid COTS Insertion (A-RCI) Sonar System. The A-RCI program, approved for development in June 1996, implemented a COTS-based open architecture for a submarine sonar signal processing system. A-RCI eliminated traditional system architecture that used specialized military specification and proprietary components, embracing the use of COTS and commercial standards, so that the sonar signal processing system could be upgraded without altering other existing towed array, hull array, and sphere array sonar equipment in the submarine (Guertin & Miller, 1998). As a result, the A-RCI program reduced both the time and cost of the periodic upgrade of sonar systems in submarines, providing the fleet with state-of-the-art signal processors.

The transformation from a closed system to a COTS-based open system took several years to complete and required a significant amount of money to purchase and install the new components. In order to speed up the transition process, the A-RCI adapted a four-phase implementation strategy (Guertin & Miller, 1998). In each phase, legacy system functionality was partially replaced by COTS processors and displays. In Phase I and Phase II, A-RCI developed a Multi-Purpose Processor (MPP) to process the data from both a towed array and a hull array. Phase III added Spherical Array MPP (SA_{MPP}) and Switch MPP ($Switch_{MPP}$) to replace the legacy system spherical array processing functions. Phase IV integrated another high-frequency sail array MPP into A-RCI. By the end of Phase IV, a COTS-based open-architecture A-RCI system completely replaced the original custom system.

In order to exercise the model described in Model Development, we will use the model to study the life-cycle cost difference between two A-RCI scenarios involving the first two phases of the A-RCI architecture. In this case study, we analyze the cost trade-offs of including Phase II in the A-RCI implementation strategy. We compare the total life-cycle cost difference with and without Phase II to see which one is more beneficial (from a life-cycle cost viewpoint). Figure 3 shows the configuration of the architectures in the first three phases of the A-RCI. During the transition from Phase I to Phase II, no new functionality is added to A-RCI; the primary change was the hardware/software configuration in the Multi-Purpose Processor for towed array and hull array ($TAHA_{MPP}$). Only the customized components in the $TAHA_{MPP}$ were replaced by COTS in Phase II to increase data processing capability. If Phase II had been integrated with Phase III, the partial transition cost from Phase I to Phase II could have been saved (the cost to initiate a transition could have been saved and the present values of redesign and requalification would have been less due to a non-zero WACC). On the other hand, including Phase II allowed the benefit of a COTS-based open architecture to be available earlier, thus reducing the O&S cost.

Figure 3. The First Three Phases of A-RCI Architecture

The two scenarios considered in our model are shown in Figure 4. The first scenario starts with Phase I at t_1, which is directly followed by Phase III at t_3. The second scenario follows the A-RCI phased strategy, starting with Phase I at t_1, Phase II at t_2, then Phase III at t_3.

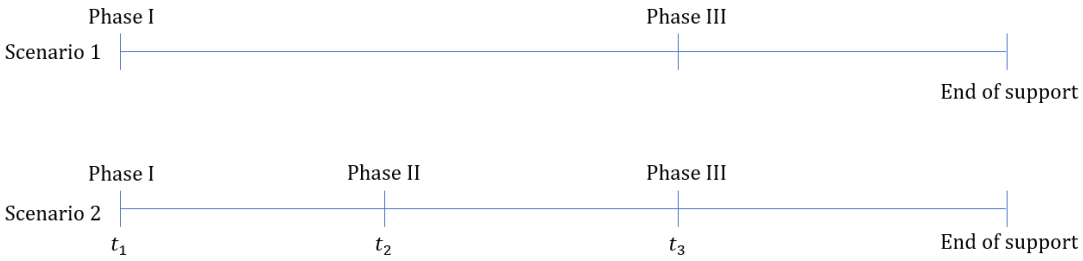


Figure 4. Scenario Description Considered in the A-RCI Case-Study Model

Figure 5 provides a conceptual demonstration of the difference in cumulative cost in both scenarios as a function of time. From t_1 to t_2 and from t_3 to end of support, in both scenarios, the systems are operated in the same architecture (i.e., Phase I and Phase III). Thus, the cost difference between the scenarios is constant before the start of Phase II and after the start of Phase III (e.g., a flat line in the graph). It should be emphasized that Figure 5 is the cumulative life-cycle cost difference between the two scenarios (not the cumulative life-cycle cost of the scenarios).

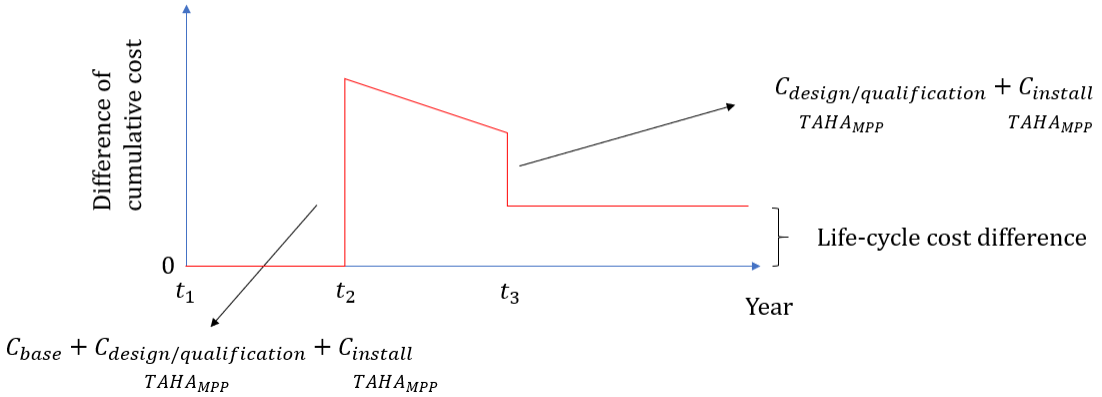


Figure 5. Life-Cycle Cost Difference Between Scenarios 1 and 2.

From t_1 to t_2, A-RCI is operated in Phase I in Scenario 1 and in Phase II in Scenario 2. Scenario 2 is taking the advantage of COTS-based open architecture in Phase II to lower O&S cost. Therefore, the cost difference is decreasing. Since Phase I and Phase II only deal with the MPP, and both phases have similar functionality, the cost difference between the two scenarios is due primarily to the degree of openness and the fraction of COTS in the implementations.

t_2 and t_3 are the times when phase transition occurs. At t_2, a phase transition cost is charged in Scenario 2. At t_3, both scenarios are charged the transition cost. The transition cost is the sum of a base cost, design/qualification costs, and installation costs. Base cost is the expense to initiate a phase transition, which typically includes the administrative cost, the penalty cost due to system downtime, etc. Design/qualification cost is related to the architecture change between the current phase and the next phase. Installation cost is the cost of purchasing and installing new equipment into the fielded systems. Based on the transition cost definition, assume that the cost of design, qualification, and installation of an MPP is independent of all the other MPPs in

the architecture. The cost difference at t_3 would only depend on the design, qualification, and installation cost of the $TAHA_{MPP}$ transition from Phase I to Phase II since the expense of SA_{MPP} and $Switch_{MPP}$ developed in Phase III is a wash. Moreover, the cost difference at t_2 is greater than at t_3 because the base cost is included in the cost difference at t_2 and is a wash between the two scenarios at t_3. If the date of implementation of Phase III (t_3) extended, Phase II has more influence and the life-cycle cost difference becomes smaller (possibly negative).

A-RCI Case Input Data

Figure 6 and Table 1 show the assumed bill of materials (BOM) for the towed array/hull array MPP and input data for the cases compared. A-RCI1 represents the architecture in Phase I, and A-RCI2 is for Phase II. The data is based on information provided in Guertin and Miller (1998) and Schramm (2013).

There are several drawers in the MPP; a signal conditioner drawer receives towed and hull array element data and conducts initial processing. The data was then forwarded to allocatable processor drawers for beamforming and further signal processing. The final data was sent out to the Control Display Workstation for remote display.

The difference between the MPP architectures in the Phase I and Phase II is the components. In Phase I, a Sun SPARC processor card is used in the signal condition drawer, which is replaced by a Motorola PowerPC processor card in Phase II. For the allocatable processor drawers, there are three custom beamforming cards and four Quad i860 cards in Phase I, which were replaced by Quad PowerPC COTS cards in Phase II. Overall, A-RCI2 is nominally more open[4] than A-RCI1.

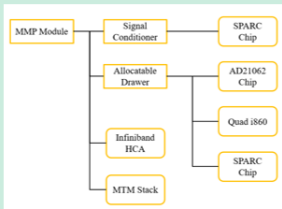
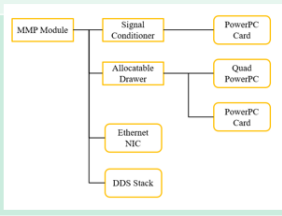
	Name	Interface Standard	Procurement Cost ($)	Procurement Life (years)	Reliability	# of instance	Architecture graph
A-RCI1	Infiniband HCA	Infiniband	$12500	Triangular(7,8,9)	Weibull(1.75,4)	1	
	MTM Stack	MTM	$5000	Triangular(7,8,9)	Weibull(1.75,4)	1	
	SPARC Card	SPARC	$9000	Triangular(7,8,9)	Weibull(1.75,2)	2	
	AD21062 Card	SHARC	$9000	Triangular(7,8,9)	Weibull(1.75,4)	1	
	Quad i860	RISC-1	$20000	Triangular(7,8,9)	Weibull(1.75,4)	2	
A-RCI2	Ethernet NIC	Ethernet	$7500	Triangular(3,4,5)	Weibull(1.75,4)	1	
	DDS Stack	DDS	$5000	Triangular(3,4,5)	Weibull(1.75,4)	1	
	PowerPC Card	RISC-2	$5500	Triangular(3,4,5)	Weibull(1.75,4)	4	
	Quad PowerPC	RISC-2	$20000	Triangular(3,4,5)	Weibull(1.75,6.1)	5	

Figure 6. A-RCI Phase I and II Architectures Assumed

[4] A-RCI2 (more open) has components that are more accessible and lower design/integration costs and production costs.

Table 1. Maintenance Strategy and Cost Assumptions[5]

Production schedule	60 systems[6] are produced in first five years (i.e., 12 systems per year)
Failure replace/repair	Good-as-new replace/repair
Average cost of maintenance per failure	$10,000/failure
Average inventory holding cost per component	$100/part/year
Spare buffer	Zero buffer: number of spare procurements would be the same as the number of failures until the next refresh/transition
Discount rate (WACC)	1%/year
Non-recurring cost of refresh	Base cost: $0.5 million/refresh Qualification cost per new component: $0.2 million Maximum non-recurring cost (entire system): $1.5 million
Cost to backfit one ship set	Total component procurement cost: $12,000 Installation cost: $3,000

In order to make sure that the cost difference from the two scenarios is solely due to the openness difference, factors that may affect life-cycle cost are isolated. For example, the renewal function (i.e., the expected number of failures and accompanied repairs in time) plays an important role in O&S cost. To minimize the effect caused by renewal function differences, A-RCI1 and A-RCI2 are assumed to have similar reliability.

Modeling Results

In order to create a valid apples-to-apples comparison of the two system management scenarios, we assume that each case is managed under its individual optimum O&S strategy. In order to accomplish this, we must first determine the optimal management of obsolescence (i.e., optimum refresh schedule, assuming bridge buys of discontinued components between refreshes).

Optimum Refresh Schedule

To compare the life-cycle cost difference between two architectures, the first step is to find each architecture's optimum refresh schedule.[7] It is inappropriate to compare life-cycle cost of the two architectures assuming the same refresh schedule. More frequent refreshing could disadvantage a closed system, which has a higher refresh cost. Conversely, fewer refreshes will increase the cost of inventory and bridge buys in an open system due to the shorter procurement life of COTS.[8]

[5] The results for the case study are highly dependent on the refresh and maintenance costs in this table. These costs were assumed and may not be representative of reality.

[6] The A-RCI program was expanded to provide improvements that could be backfit into over 60 ship sets. (Office of the Director, Operational Test and Evaluation [DOT&E], 2002).

[7] In the actual A-RCI program, the systems are refreshed every two years in order to take advantage of the advanced performance of evolving processors. In our case, the optimum strategy is selected based on a purely cost perspective.

[8] Rapid technology evolution has resulted in fast-paced product development and short product support lifetimes for COTS. COTS vendors release new products every year and are unwilling to support their products for extended periods after procurement.

In order to evaluate the optimum refresh schedule, we compared the partial life-cycle cost[9] with different refresh strategies. Figure 7 and Figure 8 show the partial life-cycle cost as a function of end-of support year for different refresh strategies. For the same architecture with different refresh strategies, the non-recurring cost such as first design cost is the same. Different refresh strategies only affect the O&S cost and refresh cost. In Figures 7 and 8, the slope of the partial life-cycle cost represents the rate of cost increase. Thus, optimum refresh strategy is selected by choosing the strategy that has the minimum average slope. For A-RCI1, the optimum strategy is to treat every obsolescence event with a lifetime buy, while a six-year refresh schedule is the optimum strategy for A-RCI2 (for the data assumed in this case study).

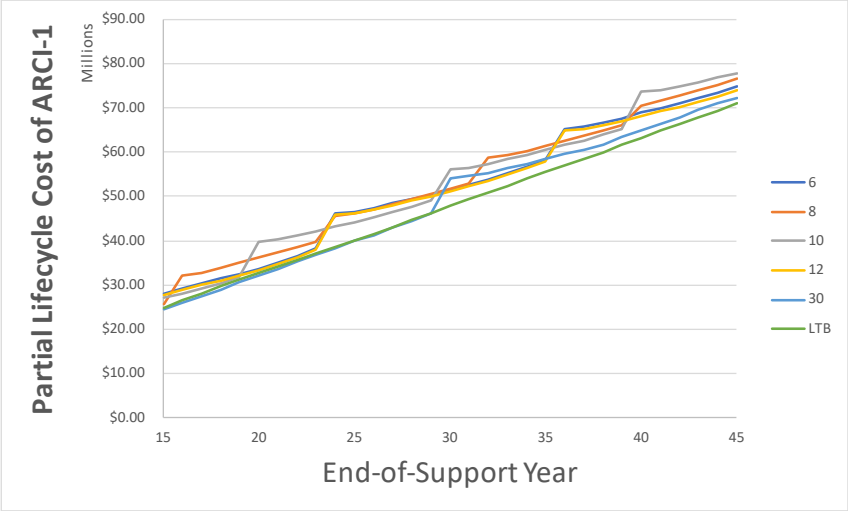


Figure 6. Partial Life-Cycle cost of A-RCI1 Given Different Refresh Strategies. LTB = all lifetime buys, no refreshes.

[9] We use the term "partial life-cycle cost" to refer to the portion of the life-cycle cost included in the model, which does not include all contributions to the life-cycle cost of the case. See the discussion in Comparing Open and Closed Systems – Relative Cost Models.

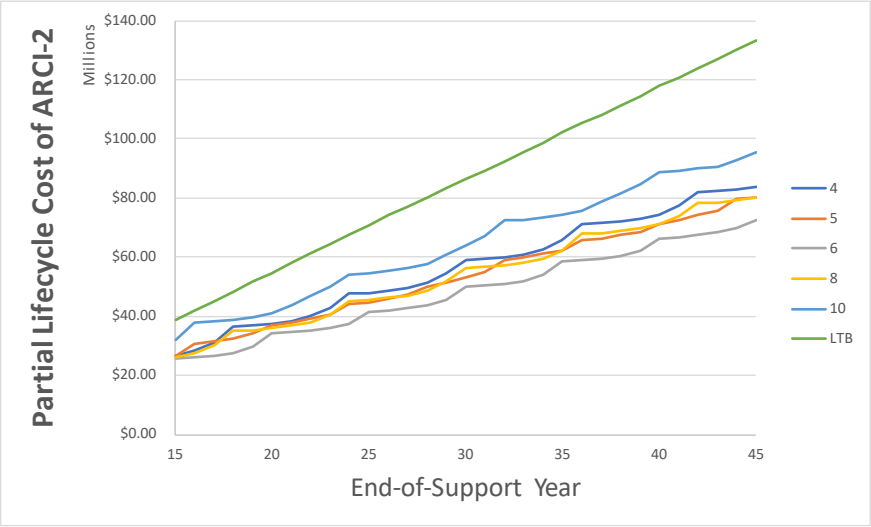


Figure 7. Partial life-cycle cost of A-RCI2 given different refresh strategies. LTB = all lifetime buys, no refreshes.

To understand the factors that influence the choice of optimum refresh strategy, we consider the composition of the relative life-cycle cost. Since the design, qualification, and production are the same for given different refresh strategies, only O&S cost and refresh cost would contribute to the difference. The increase of refresh period has an opposite effect on the increasing rate of O&S cost and refresh cost. Assume that the refresh cost for each refresh is approximately the same. For a longer refresh period, the average refresh cost per year would be less. On the other hand, more expense, such as purchasing spares and inventory management, would be spent in order to support the systems. Therefore, the weights of O&S cost and refresh cost would determine which refresh strategies are the best. For a closed system A-RCI, refresh cost is larger and dominates the total cost; therefore, a lifetime buy strategy with zero refresh cost would be the optimum.

Cost Difference Comparison Between Scenarios

After determining the optimum refresh schedule for each architecture, the two scenarios are considered. The first scenario starts with Phase I, then transitions directly to Phase III, and uses a bridge buy strategy to manage obsolescence. The second scenario follows the A-RCI phase strategy, starting with Phase I with bridge buy strategy. At year 5, the architecture transitions from Phase I to Phase II. During Phase II, the system implements a six-year refresh schedule, which occurs in years 11, 17, 23, and so on. The transition to Phase III occurs at variable dates.

Figure 9 shows the life-cycle cost difference as a function of when Phase III is implemented. The life-cycle cost difference starts at a positive value ($2 million) when the Phase III implementation year is approximately year 6 and decreases as the date of Phase III implementation increases. The negative slope of this life-cycle cost function indicates that the O&S cost of A-RCI1 in Scenario 1 is higher than A-RCI2 in Scenario 2. There are peaks at years 11, 17, 23, and 29 when refresh takes place and adds refresh cost in Scenario 2. The cost break-even point of the two scenarios occurs when the implantation year is between 12 to 13. If A-RCI implements Phase III before year 12, the difference (Scenario 2 – Scenario 1) is positive, indicating that the cost avoidance of A-RCI 2 is smaller than the implementation cost of Phase II, so Scenario 2 is more expensive. By contrast, if A-RCI implements Phase III after year 13, the life-cycle cost difference becomes negative, indicating that the cost avoidance in O&S from Phase

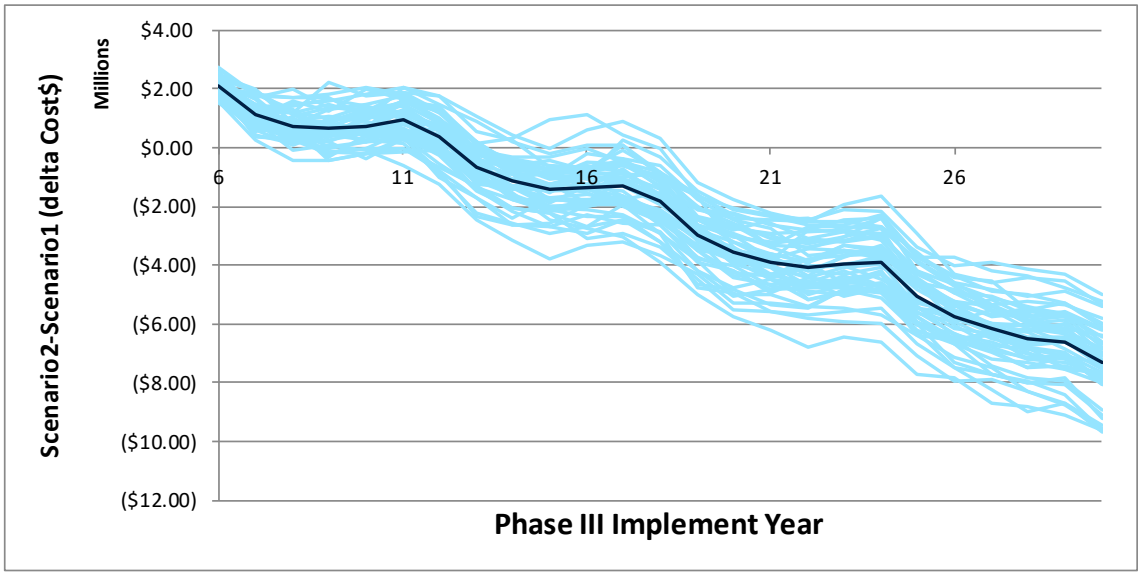


Figure 8. Life-Cycle Cost Difference Analysis Result as a Function of the Phase III Implementation Date. 50 time-history results are shown.

II (i.e., COTS-based open system) has paid off.[10] In short, the cost difference is dominated by transition cost difference before year 12 and is dominated by O&S cost difference after year 12 since the O&S cost difference grows faster. For instance, since Scenario 1 has a smaller transition cost but larger O&S cost, Scenario 1 has a lower cost before year 12 but higher cost after year 12. This simulation result indicates that the utility of Phase II (from a purely cost perspective) depends on the implementation schedule for Phase III. It shows that after the implementation of Phase II, it takes at least seven years for Phase II to pay off (from a life-cycle cost perspective). In this case, when Phase I and II are implemented in year 0 and 5 respectively, Phase III should be introduced after year 12 so that the A-RCI can benefit from Phase II architecture. In other words, if A-RCI is required to introduce Phase III before year 12 and Phase II is implemented in year 5, we should choose the implementation strategy in Scenario 1 and eliminate the option of Phase II.

Based on the simulation results, the probability distribution of the cost difference for a given Phase III implementation year can be calculated. Figure 10 is an example of when Phase III is implemented in year 13. From this distribution, the mean value of the cost difference is approximately $648,000, and the sample standard deviation is $761,000. The probability that the cost difference is less than zero is approximately 82%, indicating that if Phase III is implemented in year 13, there is an 82% probability that Scenario 2 is more beneficial than Scenario 1.

Cost Difference Comparison With Different Number of Systems

In the previous section, we considered 60 systems manufactured in the first five years. In this section, we focus on how the number of systems manufactured and fielded influences the value of the cost difference between the two scenarios. In the model, we considered 40, 60, and 80 systems manufactured and fielded in the first five years. Figure 11 shows the life-cycle cost difference result with different numbers of systems. Each curve is the mean value of 50 time-history results. According to the graph, the cost break-even points of the three curves are all between year 12 and 13 years. 80 systems has the highest positive value at year 6 and has the

[10] This is NOT a general result (it is an artifact of the data assumed).

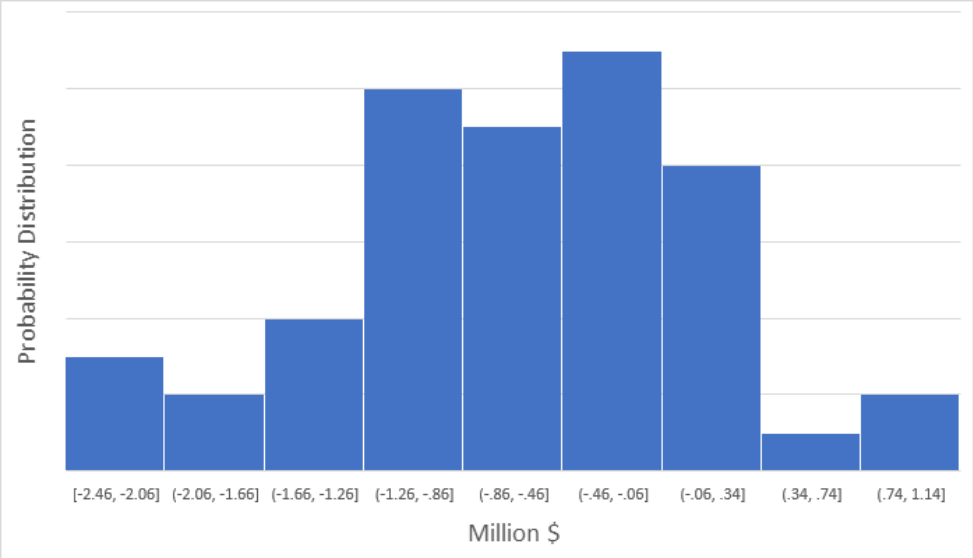


Figure 9. Probability Distribution of Life-Cycle Cost Difference When Phase III Is Implemented in Year 13

highest rate of decrease in the years that follow. More systems lead to higher administrative cost during phase transition. In year 5, the transition cost for more systems is higher in Scenario 2 and stays zero in Scenario 1 (since there is no transition in Scenario 1). Therefore, increasing the number of systems enhances the cost difference in year 5. In addition, with more fielded systems, more failures would occur, and the O&S cost difference between A-RCI1 and A-RCI would be larger as well, resulting in a steeper curve in Figure 11.

Discussion and Conclusions

It is often taken for granted that the use of open system architecture (OSA) decreases the total life-cycle cost of a system. However, there is a lack of studies quantifying the cost avoidance and assessing the circumstances under which this assumption is true. This paper presents a

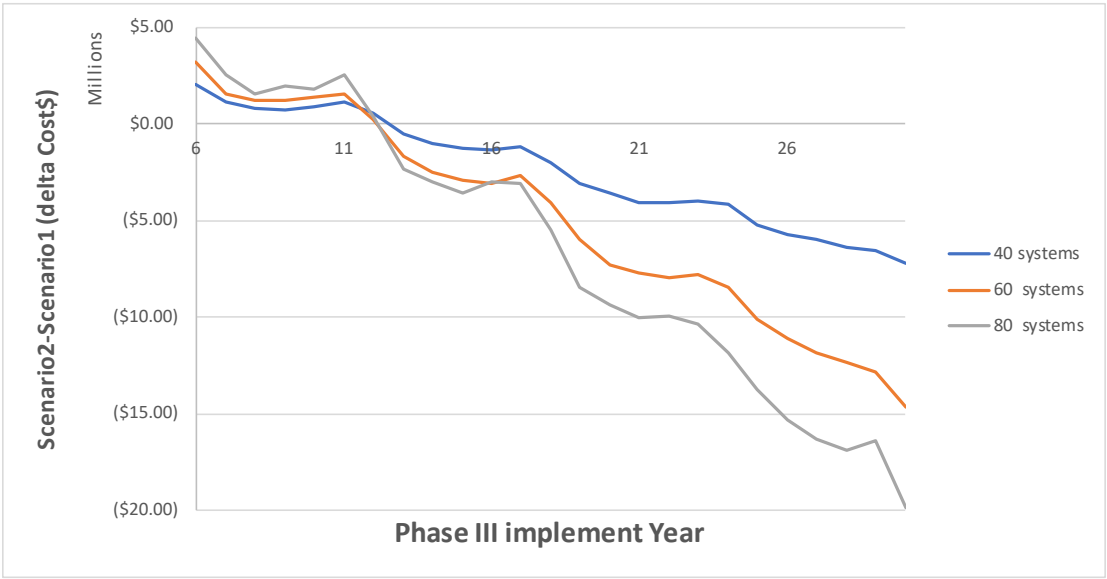


Figure 10. Comparison of Life-Cycle Difference With Different Numbers of Fielded Systems

framework for quantitative analysis of OSA by modeling life-cycle cost difference associated with system openness, including open architecture, open standard, and commercial off-the-shelf (COTS). The cost impact of openness is evaluated by converting these concepts of openness into lifetime events and corresponding costs.

A stochastic discrete-event simulation model was developed to determine the difference in life-cycle cost between two versions of the same system. The model generates events based on system information (BOM, architecture, reliability, etc.), and the consequent cost of each event is then calculated. The sums of total event cost in each of the two versions are differenced to determine which is more beneficial. This model can calculate the cost avoided and added due to openness. The simulation results can also be applied to management strategy optimization.

An A-RCI case study has been used to demonstrate the application of quantitative analysis on cost in relation to system openness. The life-cycle cost difference between two scenarios (with and without A-RCI Phase II) was evaluated as a function of the A-RCI Phase III implementation date. It should be noted that the results presented in this paper are preliminary and highly dependent on the input data, which may not reflect reality.

Future work will include several elements. First, more openness impacts should be considered in the model, for example, the impact of open source on software. It is believed that open source reduces the development cost since a wider community of developers are able to review and test the code. However, when the source code is published openly, hackers can easily find and exploit vulnerabilities in the software.[11] Thus, the tradeoff is that more money would need to be spent on cybersecurity enhancement. Second, further study will focus on developing the relationship between system openness and life-cycle cost. A model of life-cycle cost associated with COTS Functional Density has been proposed from a software perspective (Abts, 2002). For a complex system integrating both hardware and software, a more sophisticated model is needed. Moreover, since system openness is not just about the number of COTS, a quantitative metric for system openness should also be developed.

Acknowledgement

Funding for this work is provided by Naval Postgraduate School (Grant Number HQ00341910006).

References

Abbott, J. W., Levine, A., & Vasilakos, J. (2008). Modular/open systems to support ship acquisition strategies. In *Proceedings of the American Society of Naval Engineers Day*. Arlington, VA.

Abts, C. (2002). COTS based systems (CBS) functional density: A heuristic for better CBS design. In *COTS-based software systems* (pp. 1–9). Orlando, FL: Springer.

Bass, L., et al. (2008, July). *Results of SEI independent research and development project* (Technical Report CMU/SEI-2008-TR-017). Software Engineering Institute, Carnegie Mellon University.

Boudreau, M. (2006, October 30). *Acoustic rapid COTS insertion: A case study in spiral development*. Monterey, CA: Naval Postgraduate School.

Boudreau, M. (2007, April). Acoustic rapid COTS insertion: A case study in modular open systems approach for spiral development. In *Proceedings of the International Conference on Systems of Systems Engineering*.

[11] Consequently, it is believed that proprietary software is better protected against external attacks, making it more secure. Experience shows that when open-source code is actively reviewed, it has proven to be secure (take the popular operating system Linux, for example; Taylor, 2018). However, this may not be the case with software that has a limited distribution.

Clark, B., & Clark, B. (2007, June). Added source of costs in maintaining COTS-intensive systems. *Cross Talk, the Journal of Defense Software Engineering, 20*(6), 4–8.
Congressional Budget Office. (2019). *Long-term implications of the 2020 future years defense program.*
DoD. (2015, January 7). *Operation of the defense acquisition system* (DoD Instruction 5000.02). Retrieved from https://www.acq.osd.mil/fo/docs/500002p.pdf
DoD. (2018, August 31). *The defense acquisition system* (DoD Directive 5000.01). Retrieved from http://acqnotes.com/wp-content/uploads/2014/09/DoD-Directive-5000.01-Defense-Acquisition-System-31-Aug-2018.pdf
Firesmith, D. (2015, October 19). Open system architectures: When and where to be closed. *Software Engineering Institute Blog*. Retrieved from https://insights.sei.cmu.edu/sei_blog/2015/10/open-system-architecture-when-and-where-to-be-closed.html
GAO. (2014, June 26). *Review of private industry and Department of Defense open systems experiences* (GAO-14-617R).
Guertin, N. H., & Miller, R. W. (1998). A-RCI: The right way to submarine superiority. *Naval Engineers Journal, 110*(2), 21–33.
Hanratty, J. M., Lightsey, R. H., & Larson, A. G. (2002). *Open systems and the systems engineering process*. Office of the Under Secretary of Defense for Acquisition and Technology, Open Systems Joint Task Force.
Henderson, P. (2009, December 14). The case for open systems architecture. Retrieved from http://pmh-systems.co.uk/Papers/MOSACaseFor/
Jensen, F., & Petersen, N. E. (1982). *Burn-in: An engineering approach to the design and analysis of burn-in procedures*. New York, NY: Wiley.
Lewis, P., Hyle, P., Parrington, M., Clark, E., Boehm, B., Abts, C., & Manners, R. (2000). *Lessons learned in developing commercial off-the-shelf (COTS) intensive software systems* (Federal Aviation Administration Software Engineering Resource Center Report).
Logan, G. T. (2004, May). *The modular open systems approach (MOSA)* [OSJTF presentation to the Executive Program Managers Course] Retrieved from http://acc.dau.mil/CommunityBrowser.aspx?id=37585
MOSA. (2012). *AFRL/RYM metrics working group, MOSA metrics calculator*. Unpublished.
National Defense Authorization Act for Fiscal Year 2017, Pub. L. No. 114–328, 10 U.S.C. § 2446a, 2016.
National Defense Authorization Act or Fiscal Year 2020, Pub. L. No. 116–92, 10 U.S.C. § 840, 2019.
Naval Open Architecture Enterprise Team. (2009). *Open architecture assessment tool: User's guide* (Ver. 3.0).
Office of the Director, Operational Test and Evaluation. (2002). *2002 annual report*. Retrieved from https://www.dote.osd.mil/Publications/Annual-Reports/2002-Annual-Report/
Open Systems Joint Task Force. (2004, September). *Program manager's guide: A modular open systems approach (MOSA) to acquisition*. DoD. Retrieved from http://www.acqnotes.com/Attachments/Program%20Managers%20Guide%20to%20Open%20Systems,%20Sept%202004.pdf
Sandborn, P. (2017). *Cost analysis of electronic systems* (2nd ed.). World Scientific.
Schramm, Z. (2013). *A model for estimating the cost tradeoffs associated with open electronic systems* (Master's thesis, Department of Mechanical Engineering, University of Maryland).
Singh, P., & Sandborn, P. (2006, April–June). Obsolescence driven design refresh planning for sustainment-dominated systems. *The Engineering Economist*, *51*(2), 115–139.
Taylor, D. (2018, February 6). Why Linux is better than Windows or macOS for security. *Computerworld*. Feb, 6, 2018. Retrieved from

https://www.computerworld.com/article/3252823/why-linux-is-better-than-windows-or-macos-for-security.html
Wright, M., Humphrey, D., & McCluskey, P. (1997, June). Uprating electronic components for use outside their temperature specification limits. *IEEE Transactions on Components, Packaging, and Manufacturing Technology* (Part A), *20*(2), 252–256.

Acquisition Research Program
Graduate School of Defense Management
Naval Postgraduate School
555 Dyer Road, Ingersoll Hall
Monterey, CA 93943

www.acquisitionresearch.net